logically dividing the memory into a plurality of heaps, each heap dedicated to one processor for garbage collection;

performing a plurality of garbage collection phases, wherein each processor having a dedicated heap, each processor performs each of the phases on the heap dedicated to the processor using a garbage collection thread executing on the processor; and

synchronizing the processors so that all processors have completed the preceding phase prior to each processor beginning the next phase.

2. (Original) A method as defined in claim 1 wherein the synchronizing act comprises:

for each processor performing a phase of the garbage collection process, upon completion of the phase of the garbage collection process waiting for the other processors to complete the phase of the garbage collection process; and

once the other processors have completed the phase of the garbage collection process, beginning the next phase of the garbage collection process.

3. (Original) A method as defined in claim 2 wherein the garbage collection phases further comprise:

a marking phase that marks all reachable objects in memory;

a planning phase that plans the relocation of the objects;

a relocation phase that updates the object references based on information calculated by the planning phase; and

a compaction phase that compacts the reachable objects in memory.

4. (Original) A method as defined in claim 3 wherein the planning phase maintains a directory of object references and wherein the relocation phase further comprises:

analyzing each memory object to retrieve references to other memory objects;

if a reference to another memory object is present, analyzing the reference information to determine which heap the referenced object is associated;

analyzing the directory of the heap for the referenced object to determine a new address location of the referenced object; and

updating the reference information in the memory object.

5. (Original) A method of performing garbage collection in a computer system having a shared memory and a plurality of processing units, wherein the memory is divided into heaps and each heap is associated with a single processing unit, said method comprising:

stopping executing process threads;

initiating parallel marking threads in each processing unit associated with a heap, wherein one thread executes within each processing unit and wherein the marking threads mark the reachable objects in the shared memory;

upon completion of all marking threads, initiating parallel planning threads in each processing unit associated with a heap, wherein one thread executes within each processing unit and wherein each planning thread plans the new locations for objects within the associated heap;

upon completion of all the planning threads, initiating parallel relocating threads in each processing unit associated with a heap, wherein one thread executes within each processing unit and wherein each relocating thread updates internal object references

based on the new locations determined by the planning threads, the relocation threads updating information for objects within the associated heap; and

upon completion of all the relocating threads, initiating parallel compacting threads in each processing unit associated with a heap, wherein one thread executes within each processing unit and wherein each compacting thread updates moves objects within the associated heap to the new locations determined by the planning threads.

6. (Original) A method as defined in claim 5 wherein the planning phase maintains a directory of object references and wherein the relocation phase further comprises:

analyzing each memory object to retrieve references to other memory objects;

if a reference to another memory object is present, analyzing the reference information to determine which heap the referenced object is associated;

analyzing the directory of the heap for the referenced object to determine a new address location of the referenced object; and

updating the reference information in the memory object.

7. (Original) A method as defined in claim 5 wherein the marking threads mark objects independently of the heap boundaries.

8. (Original) A method as defined in claim 5 wherein all the processing units associated with the computer system are associated with a heap.

9. (Original) A method as defined in claim 5 wherein the heaps comprise a contiguous set of memory objects within the shared memory.

10. (Original) A system for performing garbage collection in a shared memory environment, the shared memory being accessed by a plurality of processing units, the

shared memory divided into heaps and each heap is associated with one processing unit, the system comprising:

for each processing unit associated with a heap:

a marking module executing a marking phase that marks reachable objects within the shared memory;

a planning module for executing a planning phase that plans the relocation the memory objects within the associated heap following the marking of all reachable objects;

a relocating module for executing a relocating phase that updates the object references within objects of the associated heap following the planning of the relocation;

a compacting module for executing a compacting phase that moves the memory objects of the associated heap following the updating of the object references; and

a rendezvous module for determining whether all processing units in the system have completed each preceding phase before starting the next phase.

11. (Original) A computer program product readable by a computer and encoding instruction for executing a computer process for collecting garbage in a computer system having a memory and a plurality of multiprocessors that share the memory, the process comprising:

logically dividing the memory into a plurality of heaps, each heap dedicated to one processor for garbage collection;

performing a plurality of garbage collection phases, wherein each processor having a dedicated heap performs each of the phases using a garbage collection thread executing on the processor; and

synchronizing the processors so that each processor has completed the preceding phase prior to beginning the next phase.

12. (Original) A computer program product as defined in claim 11 wherein the synchronizing act comprises:

for each processor performing a phase of the garbage collection process, upon completion of the phase of the garbage collection process waiting for the other processors to complete the phase of the garbage collection process; and

once the other processors have completed the phase of the garbage collection process, beginning the next phase of the garbage collection process.

13. (Original) A computer program product as defined in claim 12 wherein the garbage collection phases further comprise:

a marking phase that marks all reachable objects in memory;

a planning phase that plans the relocation of the objects;

a relocation phase that updates the object references based on information calculated by the planning phase; and

a compaction phase that compacts the reachable objects in memory.

14. (Original) A computer program product as defined in claim 13 wherein the planning phase maintains a directory of object references and wherein the relocation phase further comprises:

analyzing each memory object to retrieve references to other memory objects;

if a reference to another memory object is present, analyzing the reference information to determine which heap the referenced object is associated;

analyzing the directory of the heap for the referenced object to determine a new address location of the referenced object; and

updating the reference information in the memory object.

15. (Original) A computer program product readable by a computer and encoding instruction for executing a computer process for performing garbage collection in a computer system having a shared memory and a plurality of processing units, wherein the memory is divided into heaps and each heap is associated with a single processing unit, said process comprising:

stopping executing process threads;

initiating parallel marking threads in each processing unit associated with a heap, wherein one thread executes within each processing unit and wherein the marking threads mark the reachable objects in the shared memory;

upon completion of all marking threads, initiating parallel planning threads in each processing unit associated with a heap, wherein one thread executes within each processing unit and wherein each planning thread plans the new locations for objects within the associated heap;

upon completion of all the planning threads, initiating parallel relocating threads in each processing unit associated with a heap, wherein one thread executes within each processing unit and wherein each relocating thread updates internal object references based on the new locations determined by the planning threads, the relocation threads updating information for objects within the associated heap; and

upon completion of all the relocating threads, initiating parallel compacting threads in each processing unit associated with a heap, wherein one thread executes within each processing unit and wherein each compacting thread updates moves objects within the associated heap to the new locations determined by the planning threads.

16. (Original) A computer program product as defined in claim 15 wherein the planning phase maintains a directory of object references and wherein the relocation phase further comprises:

analyzing each memory object to retrieve references to other memory objects;

if a reference to another memory object is present, analyzing the reference information to determine which heap the referenced object is associated;

analyzing the directory of the heap for the referenced object to determine a new address location of the referenced object; and

updating the reference information in the memory object.

17. (Original) A computer program product as defined in claim 15 wherein the marking threads mark objects independently of the heap boundaries.

18. (Original) A computer program product as defined in claim 15 wherein all the processing units associated with the computer system are associated with a heap.

19. (Original) A computer program product as defined in claim 15 wherein the heaps comprise a contiguous set of memory objects within the shared memory.

20. (Original) A runtime environment for a multiprocessor system, the multiprocessor system having a plurality of processing units and a shared memory, the shared memory divided into a plurality of heaps wherein each heap is dedicated to one processing unit; the runtime environment comprising: